```
####【置顶】相关资料与网站汇集
ChainBook/IPFS-For-Chinese
IPFS指南~知乎专栏
####【置顶】文档
Kademlia协议原理简介.pdf
####【置顶】TODO:
####【置顶】<u>IPFS-词汇表</u>
####【置顶】<u>IPFS-常用调试命令</u>
####【置顶】ChainBook/IPFS-For-Chinese/IPFS Commands(ipfs命令行简介)
####【置顶】chinanf-boy/go-ipfs-zh(go-ipfs中文翻译)
####ipfs读源码
1.参数解析
比如终端上执行ipfs daemon
root.go init() 准备了Root.Subcommands/RootRO.Subcommands 包含大部分command
ipfs.go init() localCommands 准备了command: daemon/init/commands;
main.go
  main
     mainRet
        setupInterruptHandler 信号接收和处理、填充context
          cli.Run (run.go)
             Parse()->parse() (parse.go) 填充Request的Command、Arguments、Options等成
员
             makeExecutor (main.go) 准备cmdDetails(默认值初始化在ipfs.go中)、
http.client(如果需要的话),返回cmds.Executor
             exctr.Execute (executor.go) 执行具体cmd的preRun/PostRun/Run
2.daemon启动流程
daemonFunc (daemon.go) 准备core.BuildCfg、
  fsrepo.Open 准备文件锁 ~/.ipfs/repo.lock
  core.NewNode (daemon.go) 准备lpfsNode、
     setupNode (builder.go) 准备BlockStore/gcBlockstore/
       startOnlineServices (core.go)
          startOnlineServicesWithHost 准备BitSwapNetwork/
             routingOption 启动routing service
             bitswap.New (bitswap.go)
                go bs.wm.Run (wangtmanager.go)
                startWorkers (workers.go)
                  providerQueryManager (e := <-bs.findKeys)
                  taskWorker
                  rebroadcastWorker (bs.findKeys <- &blockRequest)
                  provideCollector
                  provideWorker
        bserv.New (builder.go) 准备BlockService
        dag.NewDAGService 准备dagService
        pin.LoadPinner
        resolver.NewBasicResolver
        startLateOnlineServices
  printSwarmAddrs (daemon.go)
  serveHTTPApi
  mountFuse 如果有需要的话
  serveHTTPGateway
3.get file流程
####有关NAT
P2P技术简介 - 崇山峻岭 - 博客园
检测当前网络NAT类型: https://github.com/jtriley/pystun
<u>检测nat类型流程图</u>
NAT原理与NAT穿透 - 深之JohnChen的专栏 - CSDN博客
<u>P2P技术详解(一): NAT详解——详细原理、P2P简介-网络编程/专项技术区 - 即时通讯开发者社</u>
区!
UDP打洞(成功打洞) - HotIce0 - CSDN博客
####bitswap
Bitswap原理及实现
####客户端模式DHT路由(--routing=dhtclient)、及其他实验阶段特性
https://github.com/ipfs/go-ipfs/blob/master/docs/experimental-features.md
https://github.com/chinanf-boy/go-ipfs-zh/blob/master/docs/experimental-features.zh.md
####storj白皮书-英文版
storj-whitepaper-v3
from: https://github.com/storj/whitepaper
####有关白皮书的部分解析
Understanding the IPFS White Paper part 1
####
IPFS世界的物流系统: libp2p - IPFS_Newb的博客 - CSDN博客
####
go-libp2p 入门之 PingService example
####libp2p原理及核心组件
<u>比肩IPFS和Filecoin,为什么说libp2p将成为区块链和物联网的基础设施?</u>
####有关数据存储细节
IPFS内部原理入门(An Introduction to IPFS) - yichigo的专栏 - CSDN博客
IPFS的存储与读取 - 区块网
####集群搭建
IPFS系列 集群搭建 三
####有关Kademlia协议
Kademlia协议原理简介.pdf
####有关内存精简的初期调研
1.Limiting the amount of memory go-ipfs may use (<a href="https://discuss.ipfs.io/t/limiting-the-">https://discuss.ipfs.io/t/limiting-the-</a>
amount-of-memory-go-ipfs-may-use/3211/3)
1.1 ipfs本身使用只有200-300M,但由于碎片化操作系统无法回收
                                                                                     Jun 20
           66 Quote
          So, part of this is memory fragmentation. That is, IPFS may only be using 200-300MiB but the OS can't reclaim
          some of the free memory because it's mixed in with the used memory. Unfortunately, the go language makes
          managing memory manually rather difficult so memory fragmentation is unlikely to improve.
          However, a large portion of that is probably the peerstore. We had a bug for a while where peers behind NATs
          were advertising every single ephemeral port they had used since they started. We've fixed that bug but still
          have a lot of nodes advertising these massive address lists. We're working on a couple of mitigations, but
          they're still in progress
                                                                                 Reply
1.2使用Swarm.ConnMgr.LowWater、Swarm.ConnMgr.HighWater限制节点维持连接数,从而
节省内存。目前在mac、android电视上LowWater/HighWater 默认都是600/900。
参见config说明文档: <u>https://github.com/ipfs/go-ipfs/blob/master/docs/config.md#connmgr</u>
           stebalien
                                                                                        Jun 18
           At the moment, no. Really, go-ipfs should use no more than 50-100MiB of memory and we should have a low-
            66 Quote that uses more like 25-50MiB of memory. However, we aren't there yet.
           In the meantime, you can play with the connection killing settings: Swarm. ConnMgr. LowWater and
           Swarm. ConnMgr. HighWater. These allow one to limit the number of connections your peer maintains and
           therefore the amount of memory it uses.
1.3 提到了systemd的开源软件,用于限制最大使用内存。大致看了下,systemd是要取代
initd, 称为linux启动的第一个进程的, 使用成本会很高。
    piedar
                                                                                             Jun 18
   Awesome info already from @stebalien , but I'll also mention that systemd.resource-control can help you tame
   misbehaving processes. The first thing to try is adding MemoryHigh=100M to ipfs.service (1) (if you don't have
   a ipfs.service file, start with this template 1).
    Also, you should probably run the service as a dedicated ipfs user, to protect your mom's account. Your own
    account (via ssh?) will still be able to communicate via the API endpoints on localhost.
     sudo useradd ipfs --create-home
     sudo -u ipfs ipfs init
     sudo systemctl start ipfs@ipfs && sudo systemctl enable ipfs@ipfs
     sudo loginctl enable-linger ipfs
1.4 没太懂的两个功能禁用,还不知道影响如何。
           stebalien
                                                                                           Jun 22
           You can also:
            1. Start your daemon with --routing=dhtclient to disable the DHT server.
            2. Not use the experimental badger datastore. It currently eats memory like nothing else (it's fast though).
                                                                                         Reply
1.5 ulimit 限制进程使用系统资源的工具,超限后进程被终止。
2.Measuring memory usage #5530 (<a href="https://github.com/ipfs/go-ipfs/issues/5530">https://github.com/ipfs/go-ipfs/issues/5530</a>)
ps显示使用256,pprof显示使用了76M的问题
My OS is telling me that I'm using around 256MB:
   $ ps aux | head -n1
                                                     TT STAT STARTED
                     PID %CPU %MEM
                                         VSZ
                                                RSS
                                                                           TIME COMMAND
   $ ps aux | grep ipfs
                   16719 0.5 3.1 558590036 256180 s002 Ss+
   rob
                                                                2:24pm
                                                                        6:28.99 ipfs daemon
   $
But pprof tells me that it's only using 76.38MB:
   $ curl localhost:5001/debug/pprof/heap > ipfs.heap
    % Total
               % Received % Xferd Average Speed Time
                                                          Time
                                                                   Time Current
                                   Dload Upload Total
                                                          Spent
                                                                   Left Speed
                                0 3822k
  100 78010
               0 78010
                                              0 --:--:-- 4009k
  $ go tool pprof /Users/rob/golang/bin/ipfs ipfs.heap
  File: ipfs
  Type: inuse_space
  Time: Sep 27, 2018 at 5:21pm (AEST)
  Entering interactive mode (type "help" for commands, "o" for options)
  (pprof) top5
  Showing nodes accounting for 38.29MB, 50.14% of 76.38MB total
  Showing top 5 nodes out of 207
        flat flat% sum%
                                 cum cum%
     13.47MB 17.63% 17.63% 22.97MB 30.07% gx/ipfs/QmQjMHF8ptRgx4E57UFMiT4YM6kqaJeYxZ1MCDX
      4.52MB 5.91% 44.25% 4.52MB 5.91% gx/ipfs/QmZt87ZHYGaZFBrtGPEqYjAC2yNAhgiRDBmu8Ka
4.50MB 5.89% 50.14% 4.50MB 5.89% gx/ipfs/QmYmsdtJ3HsodkePE3eU3TsCaP2YvPZJ4LoXnNk
   (pprof)
2.1内存增长的原因,碎片化,gc结束前分配内存
           Stebalien commented on 29 Sep
                                                                              Contributor
                                                                                        + \cdots ...
           Some of it will be stacks (go routines). However, those should generally only use on the order of a few
           megabytes in total.
           I have a sneaking suspicion that the rest is memory fragmentation. Basically:
             1. We allocate something and go doesn't have space for it.
             2. Go asks the kernel for a page to put this data in.
             3. Go fills up this page with a bunch of random allocations.
             4. We free a bunch of stuff but not everything in the page.
           At this point, we're not using the entire page but go can't give it back to the OS. Go will use the rest of
           the space in these pages, but we may not need them (and/or the gaps in allocations are too small to be
           useful).
           This is exacerbated by:
             1. Go-ipfs's bursty allocation patterns. We'll often do a bunch of work all at once, allocate a bunch of
               tiny things, and then free them. Unfortunately, if we also happen to allocate a long-lived object in
               one of these pages full of short-lived objects, we'll have to keep the whole page around even after
               we free the short-lived objects. High performance systems with allocation patterns like this will
               actually use separate allocators/memory pools to keep these kinds of allocations on separate
               pages. Unfortunately, go doesn't really support custom allocators.
             2. Garbage collection. I'm guessing (I haven't confirmed this) that go doesn't wait for a full GC to
               finish before asking the kernel to allocate a new page. That means when allocating and
               deallocating really quickly, we can end up asking the operating system for significantly more
               memory than we actually use at any given point in time. Unfortunately, due to (1), we may end up
               having to keep this memory reserved because we may be using small pieces of it.
           TL;DR: reduce allocation frequency and use memory pools (see Pool in sync).
3.手机app中使用ipfs方案: IPFS in mobile use case #68
(https://github.com/ipfs/notes/issues/68)
           Stebalien commented on 4 Apr
                                                                                         + •••
          So, we've introduced a low power profile that:
            1. Turns of the DHT server.
            2. Severely limits the maximum number of open connections.
          You can apply it with ipfs config profile apply lowpower.
          Many of the remaining problems are architectural and will take time to fix.
在源码中搜lowpower找到相关配置内容及使用说明:
       "lowpower": {
           Description: `Reduces daemon overhead on the system. May affect node
149 functionality - performance of content discovery and data
 150 fetching may be degraded.
 151
           Transform: func(c *<u>Confia</u>) error {
 153 >
              c.Routing.Type_= "dhtclient"
 154 >
              c.<u>Reprovider</u>.I<mark>n</mark>terval = "0"
 155
 156 >
              <u>c</u>.Swarm.<u>ConnMgr</u>.<u>LowWater</u> = 20
              c.Swarm.ConnMgr.HighWater = 40
 158 >
              c.Swarm.<u>ConnMgr.GracePeriod</u> = time.Minute.String()
 159 >
 160 >
 161 >
 162
 NORMAL > SPELL > +0 ~0 -0 master > 1:profile.go
跟pc上默认配置相比,不仅降低连接节点数,延长免疫期限,type也有basic变成了dhtclient。
125
        'Swarm": {
 126
         "AddrFilters": null,
 127
         "ConnMgr": {
 128
           "GracePeriod": "20s",
 129
           "<u>HighWater</u>": 900,
 130
           "<u>LowWater</u>": 600,
 131
           "Type": "basic"
 132
 133
         "<u>DisableBandwidthMetrics</u>": false,
 134
         "<u>DisableNatPortMap</u>": false,
 135
         "<u>DisableRelay</u>": false,
 136
         "EnableRelayHop": false
 137
 138 }
 NORMAL > SPELL >> 1:config
 "~/.ipfs/config" 138L, 5296C
使用说明:
26 To better support low-power devices, we've added a low-power config profile.
27 This can be enabled when initializing a repo by running `ipfs init` with the
28 `--profile=lowpower` flag or later by running `ipfs config profile apply lowpower`.
 29
4.Low Power Mode #4137 (https://github.com/ipfs/go-ipfs/issues/4137)
           Kubuxu commented on 11 Aug 2017 • edited ▼
                                                                               Member
           First step would be probably running go-ipfs in DHT client mode:
           ipfs daemon --routing=dhtclient
           We are working on connection closing which should further reduce the number of connections thus
           reducing bandwidth and power usage.
            9 3
          whyrusleeping commented on 16 Aug 2017
                                                                              Member
                                                                                       + 🚥
          Yeah, as @Kubuxu pointed out you probably want to use --routing=dhtclient . Its probably also
          worth turning off the reprovider with: ipfs config Reprovider.Interval "0" so your node doesnt
          rebroadcast all this things it has every 12 hours.
lowpower中的配置中包含了这两项配置内容。
5.Sweet-IPFS(https://discuss.ipfs.io/t/sweet-ipfs-full-ipfs-node-for-android/3779/16)
app代码: <a href="https://github.com/RHazDev/Sweet-IPFS/tree/master/app">https://github.com/RHazDev/Sweet-IPFS/tree/master/app</a>
一个文件分享软件,使用的是kotlin语言写的,非常简捷的几个文件,可以看到使用的就是上面
提到的lowpower模式的配置。
在电视上安装了apk,很多功能都未实现。
 Thread{
       val exec = run("init")
       Thread {
            exec.inputStream.bufferedReader().forEachLine { println(it) }
       }.start()
       Thread {
            exec.errorStream.bufferedReader().forEachLine { println(it) }
       }.start()
       exec.waitFor()
       config{
            it.getAsJsonObject("Swarm").getAsJsonObject("ConnMgr").apply {
                 remove("LowWater")
                 addProperty("LowWater", 20)
                 remove("HighWater")
                 addProperty("HighWater", 40)
                 remove("GracePeriod")
                 addProperty("GracePeriod", "120s")
            }
       progress.dismiss()
       act.runOnUiThread(callback)
  }.start()
6.textile-go(https://github.com/textileio/)
一个图片分享软件,提供android app使用的aar库文件,代
码: https://github.com/textileio/textile-mobile
7.IPFS light clients(https://github.com/ipfs/notes/issues/206)
iphonex上能耗问题
使用lowpower模式有所好转但仍然很高
           magik6k commented on 3 Apr • edited -
                                                                                       + \cdots
                                                                              Member
           @carblock, can you try running the latest go-ipfs master with low-power profile - ipfs init
           --profile=lowpower or ipfs config profile apply lowpower (https://github.com/ipfs/go-ipfs/blob
           /master/repo/config/profile.go#L78) and see how/if it improves things for you?
           carblock commented on 11 Apr
                                                                                       + 😬
          Hey Łukasz,
           Thank you for the suggestion. Let you know that the energy impact is improved, though still high, after
           applying the low-power profile.
           IPFSDemo PID 3962
                             Energy
           Memory
                                        Average Energy Impact
                                                                      Average Component Utilization
                                             High
                                                                          CPU
2.3%
                                             Energy Impact
                             Energy Impact
                                         16:55.000 17:00.000 17:05.000 17:10.000 17:15.000
提到有dht相关问题fix,以及bitswap优化改善不少。(https://github.com/ipfs/go-
ipfs/issues/3786)
          Stebalien commented on 7 Aug
                                                                                    + \cdots
          I believe the last two versions have some DHT fixes that should lead to fewer spurious DHT requests to
          DHT nodes running in client-only mode. That should drive down power usage a bit.
          One of the biggest consumers is bitswap so fixing ipfs/go-ipfs#3786 should also help a lot.
8. The minimum requirements for the server are configured (<a href="https://github.com/ipfs/go-">https://github.com/ipfs/go-</a>
```

ipfs/issues/4145)

####windows平台编译/运行

####有关mac、电视上环境搭建

【IPFS + 区块链 系列】 入门篇 - IPFS环境配置及博客搭建

打包Golang并在Android环境下运行 - CSDN博客

004_IPFS - ABPS - 乐视Wiki

go-ipfs/docs/windows.md

缓慢增长。

参考:

解释了--routing=dhtclient及ipfs config Reprovider.Strategy pinned 会有什么影响。

还有就是.ipfs/config中swarm参数配置使用,反馈是正确配置后内存使用情况好很多,但仍然在